What is claimed is:

[Claim 1] 1. A method for securing a program comprised of a plurality of interoperable components, the method comprising:

extracting information about a function of a first component of the program that is callable by at least one other component of the program;
securing the extracted information;
in response to an attempt by a second component of the program to invoke the function of the first component, validating authenticity of the second component; and
if the second component is validated, providing access to the function of the first component using the secured extracted information.

[Claim 2] 2. The method of claim 1, further comprising:

generating a signature for the second component, so as to enable authentication of the second component.

[Claim 3] 3. The method of claim 2, wherein said step of generating a signature includes generating a selected one of an Authenticode signature and an MD5 message digest.

[Claim 4] 4. The method of claim 2, wherein said step of generating a signature includes generating a hash of the second component and encrypting the hash with a private key.

[Claim 5] 5. The method of claim 4, wherein said validating step includes decrypting the hash with a public key and comparing the hash to a known value.

[Claim 6] 6. The method of claim 1, wherein said extracting step includes extracting information from an export table of the first component.

[Claim 7] 7. The method of claim 1, wherein said extracting step includes removing the function name from an export table of the first component.

[Claim 8] 8. The method of claim 1, wherein said securing step includes obscuring the function name.

[Claim 9] 9. The method of claim 1, wherein said securing step includes creating a secure export table for securing the extracted information.

[Claim 10] 10. The method of claim 1, wherein said providing step includes routing a call by the second component to the function of the first component.

[Claim 11] 11. The method of claim 1, wherein said providing step includes returning an address of the function of the first component to the second component.

[Claim 12] 12. The method of claim 1, wherein said extracting step includes extracting information about a function of the first program specified by a user.

[Claim 13] 13. A computer-readable medium having processor-executable instructions for performing the method of claim 1.

[Claim 14] 14. A downloadable set of processor-executable instructions for performing the method of claim 1.

[Claim 15] 15. A method for securing a program comprised of a plurality of modules, at least one of the modules having export information for allowing other modules to invoke its program code, the method comprising:

generating signatures for at least some of the program's modules;
as the program is loaded, validating said signatures so as to verify authenticity of respective modules of the program;
for each module having program code that may be invoked by another module, removing that module's export information;
securely storing any removed export information;
for each module having its export information removed, blocking any attempt from another module to invoke its program code if the other module cannot be authenticated; and
if the other module is authenticated, allowing the attempt to proceed using the securely stored export information.

[Claim 16] 16. The method of claim 15, wherein said generating step includes generating a selected one of an Authenticode signature and an MD5 message digest.

[Claim 17] 17. The method of claim 15, wherein said generating step includes generating a hash of a module and encrypting the hash with a private key.

[Claim 18] 18. The method of claim 17, wherein said validating step includes decrypting the hash with a public key and comparing the hash to a known value.

[Claim 19] 19. The method of claim 15, further comprising:

providing a security module for validating authenticity of a module.

[Claim 20] 20. The method of claim 19, wherein the security module includes instructions causing the security module to be initialized before other modules of the program.

[Claim 21] 21. The method of claim 19, wherein an attempt to invoke a module having its export information removed is routed to the security module.

[Claim 22] 22. The method of claim 21, wherein the security module allows the attempt to proceed if the other module making the attempt is authenticated.

[Claim 23] 23. The method of claim 15, wherein said allowing step includes returning an address of program code of the module having its export information removed if the other module is authenticated.

[Claim 24] 24. The method of claim 15, wherein said removing step includes removing export information for a particular module specified by a user.

[Claim 25] 25. The method of claim 15, wherein said removing step includes removing information from an export table.

[Claim 26] 26. The method of claim 15, wherein said securely storing step includes obscuring removed export information.

[Claim 27] 27. The method of claim 15, further comprising:

in response to an attempt to invoke program code of a given module, verifying authenticity of the given module and blocking the attempt if the given module cannot be authenticated.

[Claim 28] 28. The method of claim 15, further comprising:

after allowing the attempt to proceed, providing for subsequent attempts by the other module to invoke the program code to directly invoke the program code.

[Claim 29] 29. A computer-readable medium having processor-executable instructions for performing the method of claim 15.

[Claim 30] 30. A downloadable set of processor-executable instructions for performing the method of claim 15.

[Claim 31] 31. A system for securing a program comprised of a plurality of interoperable components, the system comprising:

a module for extracting information about a function of a first component of the program that is callable by at least one other component of the program;
a module for securing the extracted information;
a validation module for validating authenticity of a second component attempting to invoke the function of the first component; and
a security module for blocking the attempt to invoke the function of the first component if the second component cannot be authenticated.

[Claim 32] 32. The system of claim 31, wherein the validation module validates authenticity of the second component based on examining a digital signature of the second component.

[Claim 33] 33. The system of claim 31, further comprising:

a module for generating a signature for at least some components of the program, so as to enable authentication of said at least some modules.

[Claim 34] 34. The system of claim 33, wherein the module for generating generates a selected one of an Authenticode signature and an MD5 message digest.

[Claim 35] 35. The system of claim 33, wherein the module for generating generates a hash of a module and encrypts the hash with a private key.

[Claim 36] 36. The system of claim 35, wherein the validation module decrypts the hash with a public key and compares the hash to a known value.

[Claim 37] 37. The system of claim 31, wherein the security module routes the attempt to the function of the first module if the second module is authenticated.

[Claim 38] 38. The system of claim 31, wherein the security module returns an address of the function of the first module if the second module is authenticated.

[Claim 39] 39. The system of claim 31, wherein the module for extracting removes an export table entry for the function of the first module.

[Claim 40] 40. The system of claim 31, wherein the module for securing creates a secure export table including the extracted information.

[Claim 41] 41. The system of claim 40, wherein the secure export table is created without using a clear text name for the function of the first module.

[Claim 42] 42. The system of claim 40, wherein the module for securing obscures function names in the secure export table.

[Claim 43] 43. The system of claim 31, wherein the security module inserts executable code into the second module during initialization of the second module so as to direct an attempt by the second module to invoke the function of the first module to the security module.

[Claim 44] 44. The system of claim 43, wherein the security module modifies the executable code included in the second module if the second module is authenticated so as to enable the second module to directly invoke the function of the first module.

[Claim 45] 45. A method for securing an exported function of a program, the method comprising:

extracting export information about the exported function of the program;
securing the extracted export information;
intercepting an attempt to access the exported function by an importer;
authenticating the importer for determining whether to permit access to the exported function; and
if the importer is authenticated, providing access to the exported function based on the secured extracted export information.

[Claim 46] 46. The method of claim 45, wherein the importer comprises another module of the program.

[Claim 47] 47. The method of claim 45, wherein the importer comprises another program.

[Claim 48] 48. The method of claim 45, further comprising:

if the importer cannot be authenticated, blocking access to the exported function.

[Claim 49] 49. The method of claim 45, wherein said authenticating step includes authenticating the importer based on a digital signature of the importer.

[Claim 50] 50. The method of claim 45, further comprising:

generating a digital signature for at least some executable modules of the program, so as to enable authentication of said at least some executable modules.

[Claim 51] 51. The method of claim 50, wherein said generating step includes generating a selected one of an Authenticode signature and an MD5 message digest.

[Claim 52] 52. The method of claim 50, wherein said authenticating step includes validating digital signature of the importer.

[Claim 53] 53. The method of claim 45, further comprising:

authenticating a program module including the exported function before providing access to the exported function.

[Claim 54] 54. The method of claim 45, wherein said providing step includes routing a call by the importer to the exported function.

[Claim 55] 55. The method of claim 45, wherein said providing step includes returning an address of the exported function to the importer.

[Claim 56] 56. The method of claim 45, wherein said extracting step includes removing an export table entry for the exported function.

[Claim 57] 57. The method of claim 45 , wherein said securing step includes obscuring the exported function name.

[Claim 58] 58. The method of claim 45, wherein said securing step includes creating a secure export table based on the extracted export information.

[Claim 59] 59. A computer-readable medium having processor-executable instructions for performing the method of claim 45.

[Claim 60] 60. A downloadable set of processor-executable instructions for performing the method of claim 45.